AFRL-IF-RS-TR-2006-23
Final Technical Report
January 2006

LANGUAGE MEASURE FOR ROBUST OPTIMAL CONTROL

The Pennsylvania State University

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. M414

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2006-23 has been reviewed and is approved for publication

APPROVED: /s/

CARL A. DEFRANCO
Project Engineer

FOR THE DIRECTOR: /s/

JAMES W. CUSACK
Chief, Information Systems Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	JANUARY 2006	Final Sep 2001 – Sep 2005

4. TITLE AND SUBTITLE
LANGUAGE MEASURE FOR ROBUST OPTIMAL CONTROL

5. FUNDING NUMBERS
C - F30602-01-2-0575
PE - 62301E
PR - M414
TA - 00
WU - 01

6. AUTHOR(S)

Asok Ray, Travis Ortogero

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

The Pennsylvania State University
101 Technology Center
University Park Pennsylvania 16802-7000

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFSA
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

AFRL-IF-RS-TR-2006-23

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Carl DeFranco/IFSA/(315) 330-3096/ Carl.DeFranco@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 Words)*

We present the basic language measure algorithms with some preliminary definitions of terms used. Properties of the measure are presented accompanied by proofs. We then describe in detail the algorithm for generating an optimal controller with respect to the language measure, and then prove that the generation is indeed optimal. Next we describe our experimentation regarding extracting language measure parameters from real or simulated modeled scenarios. From this experiment we obtain a constructive method for getting these parameters that can be applied to other types of situations other than MICA situations, thereby validating the language measure as a viable method of measurement. Example controllers and plant models are then used to illustrate the robustness of the measure with respect to small perturbations of its input parameters, and how the optimal algorithm generates results that improve upon the non-optimal measurements.

14. SUBJECT TERMS
Mixed initiative; robust control; UAV; command and control; formal language.

15. NUMBER OF PAGES
17

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

List of Tables

1.0 Brief Review of the Language Measure

Let the plant behavior be modeled as a DFSA $G_i \equiv (Q, \Sigma, \delta, q_i, Q_m)$ where Q is the finite set of states with $|Q| = n$ excluding the dump state ([RW87] and [KG95(1)]), if any, and $q_i \in Q$ is the initial state; Σ is the (finite) alphabet of events; Σ^* is the set of all finite-length strings of events including the empty string ε; the (total) function of $\delta : Q \times \Sigma \to Q$ represents state transitions and $\delta^* : Q \times \Sigma^* \to Q$ is an extension of δ; and $Q_m \subseteq Q$ is the set of marked states.

Definition 1: A DFSA G_i, initialized at $q_i \in Q$, generates language $L(G_i) \equiv \{s \in \Sigma^* : \delta^*(q_i, s) \in Q\}$ and marked sublanguage $L_m(G_i) \equiv \{s \in \Sigma^* : \delta^*(q_i, s) \in Q_m\}$.

The language $L(G_i)$ is partitioned into the non-marked language $L^o(G_i) \equiv L(G_i) - L_m(G_i)$ and the marked language $L_m(G_i)$, consisting of event strings that, starting from the initial state $q_i \in Q$, terminate at one of the non-marked states in $Q - Q_m$ and one of the marked states in Q_m, respectively. The set Q_m of marked states is partitioned into Q_m^+ and Q_m^-, where Q_m^+ contains all *good* marked states that we desire to reach and Q_m^- contains all *bad* marked states that we want to avoid, although it may not always be possible to avoid the bad states while attempting to reach the good states. The marked language $L_m(G_i)$ can be further partitioned into $L_m^+(G_i)$ and $L_m^-(G_i)$ consisting of good and bad strings that, starting from the initial state q_i, terminate to Q_m^+ and Q_m^-, respectively.

Now we construct a signed real measure $\mu : 2^{\Sigma^*} \to \Re \equiv (-\infty, \infty)$ for quantitative evaluation of every event string $s \in \Sigma^*$ based on state-based decomposition of $L(G_i)$ into null (i.e., $L^o(G_i)$), positive (i.e., $L_m^+(G_i)$), and negative (i.e., $L_m^-(G_i)$) sublanguages of $L(G_i)$.

Definition 2: The language of all strings that start at state $q_i \in Q$, and terminate at state $q_j \in Q$, is denoted $L(q_j, q_i)$. Thus, $L(q_j, q_i) \equiv \{s \in L(G_i) : \delta^*(q_i, s) = q_j\}$.

Definition 3: The characteristic function that assigns a signed real weight to state-partitioned sublanguages is defined as: $\chi : \{L(p,q) : p, q \in Q\} \to [-1, 1]$ such that

$$\chi(q_j) \in \begin{cases} [-1, 0) \text{ if } p \in Q_m^- \\ \{0\} \quad \text{if } p \notin Q_m \\ (0, 1] \text{ if } p \in Q_m^+ \end{cases} \text{ independent of } q_i$$

Definition 4: The event cost is conditioned on the DFSA state at which the event is generated, and is defined as $\tilde{\pi} : \Sigma^* \times Q \to [0,1)$ such that $\forall q_j \in Q,\ \forall \sigma_k \in \Sigma,\ \forall s \in \Sigma^*$,

- $\tilde{\pi}[\sigma_k \,|\, q_j] \equiv \tilde{\pi}_{jk} \in [0,1)$; $\sum_k \tilde{\pi}_{jk} < 1$;
- $\tilde{\pi}[\sigma_k \,|\, q_j] = 0$ if $\delta(q_j, \sigma_k)$ is undefined;
- $\tilde{\pi}[\varepsilon \,|\, q_j] = 1$;
- $\tilde{\pi}[\sigma_k s \,|\, q_j] = \tilde{\pi}[\sigma_k \,|\, q_j]\, \tilde{\pi}[s \,|\, \delta(q_j, \sigma_k)]$.

Now we define the measure of any sublanguage of the $L(G_i)$ in terms of the signed the characteristic function χ and the non-negative event cost $\tilde{\pi}$.

Definition 5: The signed real measure μ of a singleton string set $\{s\}L(q_j, q_i) \subseteq L(G_i) \in 2^{\Sigma^*}$ is:

$\mu(\{s\}) \equiv \chi(q_j)\tilde{\pi}(s \mid q_i) \quad \forall s \in L(q_j, q_i)$.

The signed real measure of $L(q_j, q_i)$ is defined as:

$$\mu\big(L(q_j, q_i)\big) \equiv \left(\sum_{s \in L(q_j, q_i)} \mu\big(\{s\}\big) \right)$$

The signed real measure of a DFSA G_i, initialized at the state $q_i \in Q$, is defined as:

$\mu_i \equiv \mu(L(G_i)) = \sum_j \mu\big(L(q_j, q_i)\big)$.

Definition 6: The state transition cost of the DFSA is defined as a function $\pi : Q \times Q \to [0,1)$ such that $\forall q_j,\ q_k \in Q,\quad \pi(q_k \big| q_j) = \sum_{\sigma \in \Sigma : \delta(q_j, \sigma) = q_k} \tilde{\pi}(\sigma \big| q_j) \equiv \pi_{jk}$ and $\pi_{jk} = 0$ if $\{\sigma \in \Sigma : \delta(q_j, \sigma)\} = \varnothing$. The $n \times n$ state transition cost matrix, denoted as Π-matrix, is defined as:

$$\Pi = \begin{bmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{bmatrix}$$

Proposition 1: Given a state transition cost matrix $\Pi \in \Re^{n \times n}$, the operator $[I - \Pi]$ is invertible and the bounded linear operator $[I - \Pi]^{-1} \geq 0$ where the matrix inequality is implied elementwise.

Proof: It follows from Definitions 4 and 6 that:

$\|\Pi\|_\infty \equiv \max_i \sum_j \pi_{ij} = 1 - \theta$ where $\theta \in (0,1)$

Then, $[I - \Pi]^{-1}$ is invertible and is a bounded linear operator with norm $\| [I - \Pi]^{-1} \|_\infty \leq \theta^{-1}$ [NS82, p. 431].

Using Taylor series expansion, $[I - \Pi]^{-1} = \sum_{k=0}^{\infty} \Pi^k$. Each element of Π is non-negative, so each element of Π^k is also. Thus, $[I - \Pi]^{-1} \geq 0$ elementwise. ■

Wang and Ray [WR02] and Ray and Phoha [RP02] have shown that the measure $\mu_i \equiv \mu(L(G_i))$ of the language $L(G_i)$ can be expressed as: $\mu_i = \sum_j \pi_{ij} \mu_j + \chi_i$ where $\chi_i \equiv \chi(q_i)$. Equivalently, in vector notation: $\mu = \Pi \mu + \mathrm{X}$ where the measure vector $\mu \equiv [\mu_1\ \mu_2\ \cdots \mu_n]^T$ and the characteristic vector $\mathrm{X} \equiv [\chi_1\ \chi_2\ \cdots\ \chi_n]^T$. By Proposition 1, the measure vector μ is uniquely determined as: $\mu = [I - \Pi]^{-1} \mathrm{X}$.

2.0 Optimal Control Using the Language Measure

We now present the theoretical foundations of the unconstrained optimal control of discrete event systems [SL98]. Let $\mathcal{S} \equiv \{S^0, S^1, \cdots, S^N\}$ be the finite set of all supervisory control policies for the open loop plant automaton G where S^0 is the null controller (i.e., no disabled events), i.e. $L(S^0/G) = L(G)$. Therefore the controller cost matrix $\Pi(S^0) = \Pi^0 \equiv \Pi^{plant}$ is the Π-matrix of the open loop plant automaton G. For a supervisor $S^i, i \in \{1, 2, \cdots, N\}$, the control policy selectively disables certain controllable events, and therefore the following (elementwise) inequality holds: $\Pi^k \equiv \Pi(S^k) \le \Pi^0$ and $L(S^k/G) \subseteq L(G) \;\; \forall S^k \in \mathcal{S}$.

Definition 7: For any supervisor $S \in \mathcal{S}$ and any measure vector $\nu \in \Re^n$, the affine operator $T(S): \Re^n \to \Re^n$ is defined as: $T(S)\nu = \Pi(S)\nu + \mathrm{X}$

Proposition 2: $\forall S \in \mathcal{S}$, $T(S)$ is a contraction operator, and there exists a unique measure vector $\mu(S)$ such that $\mu(S) = T(S)\mu(S)$.

Proof: Since $0 \le \Pi(S) \le \Pi^0$ elementwise, and Π^0 is a contraction operator, $\Pi(S)$ is also a contraction operator. Since X is a constant vector, $T(S)$ is also a contraction. Hence, $\exists$ a unique fixed point $\mu(S) = T(S)\mu(S)$. ■

Corollary 1 to Proposition 2: The fixed point of the contraction operator $T(S^k)$ is: $\mu^k = [I - \Pi^k]^{-1}\mathrm{X}$ where $\mu^k \equiv \mu(S^k)$ and $\Pi^k \equiv \Pi(S^k)$.

Proof: The unique fixed point $\mu(S^k)$ of $T(S^k)$ satisfies the identity $\mu(S^k) = \Pi(S^k)\mu(S^k) + \mathrm{X}$. As $0 \le \Pi(S^k) \le \Pi(S^0)$ elementwise, we have $\left\| \Pi(S^k) \right\|_\infty \le \| \Pi(S^0) \|_\infty < 1$. Hence, the operator $[I - \Pi(S^k)]^{-1}$ is bounded. ■

Corollary 2 to Proposition 2: The operator $[I - \Pi^k]$ has a real positive determinant, i.e., $Det[I - \Pi^k] > 0$.

Proof: Eigenvalues of the real matrix Π^k are located within the unit circle and they appear as real or complex conjugates. Therefore, eigenvalues of $[I - \Pi^k]$ have positive real parts. So, $Det[I - \Pi^k]$ is real positive. ■

Proposition 3: Let $\mu_j^k = \min_{\ell \in \{1,2,\cdots,n\}} \mu_\ell^k$. If $\mu_j^k \le 0$ then $\chi_j \le 0$ and if $\mu_j^k < 0$, then $\chi_j < 0$.

Proof: The DFSA satisfies the identity $\mu_j^k = \sum_{\ell \in \{1,2,\cdots,n\}} \pi_{j\ell}\mu_\ell^k + \chi_j$ that leads to the inequality $\mu_j^k \ge (\sum_\ell \pi_{j\ell})\mu_j + \chi_j \Rightarrow \; (1 - \sum_\ell \pi_{j\ell})\mu_j^\ell \ge \chi_j$. The proof follows from $(1 - \sum_\ell \pi_{j\ell}) > 0$ (see Defs. 3, 4). ■

Corollary 1 to Proposition 3: Let $\mu_j^k = \max_{\ell \in \{1,2,\cdots,n\}} \mu_\ell^k$. If $\mu_j^k \ge 0$, then $\chi_j \ge 0$ and if $\mu_j^k > 0$, then $\chi_j > 0$.

Proof: The proof is similar to that of Proposition 3. ■

Proposition 4: Given $\Pi(S^0) = \Pi^0 \equiv \Pi^{plant}$ and $\mu^k \equiv [\mathbf{I} - \Pi^k]^{-1}\mathrm{X}$, let Π^{k+1} be generated from Π^k for $k \geq 0$ as follows: $\forall i, j \in \{1, 2, \cdots, n\}$, ij^{th} element of Π^{k+1} is modified as: $\pi_{ij}^{k+1} \begin{cases} \geq \pi_{ij}^k \ if \ \mu_j^k > 0 \\ = \pi_{ij}^k \ if \ \mu_j^k = 0 \\ \leq \pi_{ij}^k \ if \ \mu_j^k < 0 \end{cases}$ and $\Pi^k \leq \Pi^0 \ \forall k$. Then, $\mu^{k+1} \geq \mu^k$ elementwise and equality holds if and only if $\Pi^{k+1} = \Pi^k$.

Proof: $\mu^{k+1} - \mu^k = \left([I - \Pi^{k+1}]^{-1} - [I - \Pi^{k+1}]^{-1}\right)\mathrm{X}$

$= [I - \Pi^{k+1}]^{-1}\left([I - \Pi^k] - [I - \Pi^{k+1}]\right)[I - \Pi^k]^{-1}\mathrm{X}$

$= [I - \Pi^{k+1}]^{-1}\left(\Pi^{k+1} - \Pi^k\right)\mu^k$

Defining the matrix $\Delta^k \equiv \Pi^{k+1} - \Pi^k$, let the j^{th} column of Δ^k be denoted as Δ_j^k. Then, $\Delta_j^k \leq 0$ if $\mu_j^k < 0$ and $\Delta_j^k \geq 0$ if $\mu_j^k > 0$, and the remaining columns of Δ^k are zero vectors. This implies: $\Delta^k \mu^k = \sum_j \Delta_j^k \ \mu_j^k \geq 0$. Since $\Pi^k \leq \Pi^0 \ \forall k$, $[I - \Pi^{k+1}]^{-1} \geq 0$ elementwise, we have. $[I - \Pi^{k+1}]^{-1}\Delta^k \mu^k \geq 0 \Rightarrow \mu^{k+1} \geq \mu^k$. For $\mu_j^k \neq 0$ and Δ^k as defined above, $\Delta^k \mu^k = 0$ if and only if $\Delta^k = 0$. Then, $\Pi^{k+1} = \Pi^k$ and $\mu^{k+1} = \mu^k$. ■

Corollary 1 to Proposition 4: Let $\mu_j^k < 0$. Let Π^{k+1} be generated from Π^k by disabling controllable events that lead to the state q_j. Then, $\mu_j^{k+1} < 0$.

Proof: Since only j^{th} column of $[I - \Pi^{k+1}]$ differs from that of $[I - \Pi^k]$ and the remaining columns are the same, the j^{th} row of the cofactor matrix of $[I - \Pi^{k+1}]$ is the same as that of the cofactor matrix of $[I - \Pi^k]$, we have $Det[I - \Pi^{k+1}]\mu_j^{k+1} = Det[I - \Pi^k]\mu_j^k$. By Corollary 2 to Proposition 2, both determinants are real positive. ■

Remark 1: In Proposition 4, some elements of the j^{th} column of Π^k are decreased (or increased) by disabling (or re-enabling) controllable events that lead to states q_j for which $\mu_j^k < 0$ (or $\mu_j^k \geq 0$). ■

Proposition 5: Iteration of the algorithm in Proposition 4 leads to an optimal cost matrix Π^* that maximizes performance vector $\mu^* \equiv [I - \Pi^*]^{-1}\mathrm{X}$ elementwise.

Proof: Let there be another cost matrix $\tilde{\Pi} \leq \Pi^0$ for which $\tilde{\mu} \equiv [I - \tilde{\Pi}]^{-1}\mathrm{X}$. We will show that $\tilde{\mu} \leq \mu^*$. Starting with $\tilde{\mu} - \mu^* = [I - \tilde{\Pi}]^{-1}\,[\tilde{\Pi} - \Pi^*]\,\mu^*$, we rearrange the elements of the μ^*-vector such that $\mu^* = [\underbrace{\mu_1^* \cdots \mu_\ell^*}_{\geq 0} \mid \underbrace{\mu_{\ell+1}^* \cdots \mu_n^*}_{<0}]^T$ where $\mu_k \geq 0$ and no controllable event leading to states q_k has been disabled; and $\mu_k < 0$ for $k = \ell + 1, \ell + 2, \cdots, n$ where all controllable events leading to states q_k, $k = 1, 2, \cdots, \ell$, have been disabled. The cost matrices $\tilde{\Pi}$ and Π^* are also rearranged by columns in the order in which the μ^*-vector is arranged.

The algorithm in Proposition 4 keeps elements in the first ℓ columns of Π^* the same as those of the (open loop plant's) Π^0-matrix and decreases the elements in the last $(n-\ell)$ columns to the maximum permissible extent by disabling all controllable events. In contrast, the columns of $\tilde{\Pi}$ are reduced by an arbitrary choice. Therefore, the $(\tilde{\Pi} - \Pi^*)$-matrix, whose first ℓ columns are non-positive and last $(n-\ell)$ columns are non-negative, yields:

$\tilde{\mu} - \mu^* = [I - \tilde{\Pi}]^{-1} [\, first\ \ell\ cols \le 0 \ \mid\ last\ (n-\ell)\ cols \ge 0]\, \mu^*$ where $\mu^* = [\underbrace{\mu_1^* \cdots \mu_\ell^*}_{\ge 0} \mid \underbrace{\mu_{\ell+1}^* \cdots \mu_n^*}_{<0}]^T$.

Since $[I - \tilde{\Pi}]^{-1} \ge 0$ elementwise, we conclude that

$$\tilde{\mu} - \mu^* = \underbrace{[I - \tilde{\Pi}]^{-1}}_{\ge 0} \left(\underbrace{\sum_{j=1}^{\ell} Col_j \cdot \mu_j^*}_{\le 0} + \underbrace{\sum_{j=\ell+1}^{n} Col_j \cdot \mu_j^*}_{\le 0} \right) \le 0$$

Therefore, $\tilde{\mu} \le \mu^*$ for any choice of $\tilde{\Pi}$. ■

Proposition 6: The control policy induced by the Π^*-matrix is unique in the sense that the controlled language is most permissive (i.e., least restrictive) among all controller(s) having the best performance.

Proof: Disabling controllable event(s) leading to a state q_j with performance measure $\mu_j^* = 0$ does not alter the performance vector μ^*. The optimal control does not disable any controllable event leading to a state with zero performance. Thus, the control policy induced by the Π^*-matrix is most permissive, among all controllers with equal performance μ^*. ■

Remark 2: Propositions 5 and 6 suffice to conclude that the Π^*-matrix yields the most permissive controller with the best performance μ^*. The control policy is realized as follows:

- All controllable events leading to the states q_j where $\mu_j^* < 0$ are disabled;
- All controllable events leading to the states q_j where $\mu_j^* \ge 0$ are enabled. ■

3.0 Experimentation

The experimentation for our research was designed to resolve the issue of determining the $\tilde{\pi}$ values of a given plant model for a system. It is important to be able to mathematically or experimentally determine these values, as opposed to their being set by hand, because in general it is impossible to properly assign by hand probabilities for uncontrollable events (it is possible to assign the $\tilde{\pi}$ values for controllable events, since a controller can choose how often they wish such events to occur). For typical MICA scenarios, we used the Boeing Simulator to generate the necessary data to determine the $\tilde{\pi}$ values.

The simulation runs generates situations where events occur. Specifically, we implemented an event generator that examined the variables of the system and determined, from those variables and the current state of the system according to the plant model, what discrete event is occurring in the system. Since the Boeing simulator is a rich simulation environment, we can use it to determine the plant model's $\tilde{\pi}$ values.

Experiment Description

The basis for the experiment is this hypothesis: ***Hypothesis:*** $\tilde{\pi}$ values can be determined through event generation and statistical analysis of those events occurring over the course of a large number of simulated or actual runs.

The goal of the experiment is to show that the experimental data for these $\tilde{\pi}$ values converges to within ε of its actual value with probability δ. The experimental procedure was as follows:

1. Identify the states and events (and thus the transition table) for the system's plant model.

2. Implement event generators that take the simulator/experiment data and determine if one of the previously defined events have occurred. Then update the plant model to the new state designated by the event that occurred and the previous state.

3. Tally event occurrences in a $\tilde{\pi}$-like matrix. That is, an entry in an m * n matrix is incremented whenever the corresponding event occurs at the corresponding state.

4. Sum the values of a row *i* and then divide each element in *i* by that sum. Scale this number by $(1-\theta)$ for some pre-determined value of θ. The resulting matrix contains the $\tilde{\pi}$ values.

5. Repeatedly run experiment to continue adding to the event occurrence matrix. Over the course of numerous experiment runs, the resulting $\tilde{\pi}$ value matrix from these runs should converge.

In order to conduct the experiment, we implemented event generators that would determine, at each tick, whether or not an event in the set of possible events given the current state had occurred. For example, a event generator for a *Damaged* event would trigger if the simulator's status for a given platform indicated that the platform was damaged. Each event required an event generator. Once these generators were developed, we could conduct the experiment.

Simulation

The experiment scenario was that of two fighters and five targets. Each fighter had the same plant model, and each fighter's events were used in the tallying of events in the event occurrence matrix. Targets were randomly placed on the map and fighters emerged from a location representing their base on the map (the southeast corner). See Figure 1.

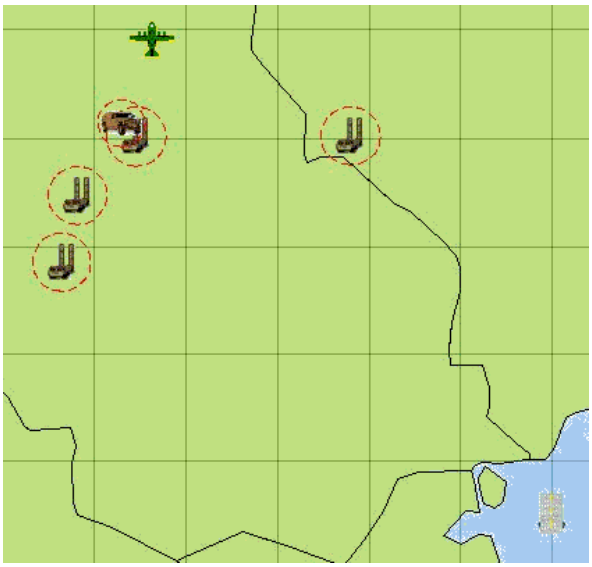

Figure 1. Sample Experiment Scenario

The fighters would emerge randomly from the base, and upon detection of a target, fly toward it to engage, and then decide, based upon uncontrollable events and preset controllable event probabilities, what actions to take. The fighter's processes continued in this manner until they were destroyed, all the targets were destroyed, or an upper limit on time for each run was reached.

Plant Model

The experiment plant model had these states:

1. AbortMission – fighter has stopped attempting combat/flight toward combat
2. AtBase – fighter is at base
3. Attack – Fighter has attacked or is attacking target
4. AttackWithDamaged – fighter is in combat and is attacking (again) with damages to itself
5. AttackWithLowFuel – fighter is in combat and is attacking (again) with low fuel
6. DamagedBeforeAttack – fighter is not in combat and has received damage
7. DamagedDuringAttack – fighter is in combat and has received damage
8. EnRoute – fighter is flying toward target/waypoint
9. LowFuelBeforeAttack – fighter is not in combat and is nearing insufficient fuel to return to base
10. LowFuelDuringAttack – fighter is in combat and is nearing insufficient fuel for returning to base
11. MidairRefuel – fighter is attempting to refuel from a tanker
12. NearTg – Fighter is within range of target
13. NoWeapons – fighter has no weapons
14. SelfDestroyed – fighter is destroyed
15. TargetDestroyed – fighter's current target has been destroyed

The following events were recognized by the generators, where U stands for uncontrollable, and C for controllable:

a. attack (C) – fighter decides to attack its current target
b. damaged (U) – fighter is damaged
c. die (U) – fighter has been destroyed
d. flee (C) – fighter told to flee to base
e. lowFuel (U) – fighter near insufficient fuel to return to base, given current position
f. midairRefuel (C) – fighter decides to refuel from a tanker
g. nearTg (U) – fighter is near its target
h. noWeapons (U) – fighter has used all available weapons (includes losing weapons to damage, jamming, etc.)
i. reachBase (U) – fighter reaches base
j. refuelCompleted (U) – fighter has finished a midair refueling
k. replan (U) – fighter has to change its course due to uncontrollable event
l. tgDestryd (U) – fighter's target has become destroyed

Transition Table

The entries of a transition table indicate the events that are possible from each state. For example, from state 1 (atBase), only one event is possible, replan, and that event takes the model to state 2 (EnRoute). Table 1 is the transition table for the plant model.

Table 1: Transition Table for Plant Model

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**
1			14						2			
2											8	
3	3	7	14	1	10			13			8	15
4	4		14	1				13			8	15
5	5		14	1				13			8	15
6			14	1			7				8	
7	4		14	1							8	
8		6	14		9		12	13			8	
9			14	1		11	10				8	
10	5		14	1		11					8	
11			14							8		
12	3		14					13			8	15
13			14	1								
14												
15			14								8	

Results

After several hundred simulation runs, the entries with frequently occurring events began to stabilize near a particular value. For example, the last few $\tilde{\pi}$ values for the transition <Attack, attack, Attack> (from the Attack state, choose to attack) are shown just below in Table 2. The resulting $\tilde{\pi}$ matrix, prior to any scaling, is in Table 3 (figures may not sum to 1 due to rounding).

0.439537
0.439553
0.43964
0.439727
0.439835
0.439878
0.439864
0.439893
0.439879
0.439843

Table 2: Value Stabilization

Note that some entries are 0, despite their having potential for having a transition occur there, according to the transition table. There are a few possible reasons for this result. One is that not enough runs might have been conducted on the model to get enough situations for those transitions to occur. Another more likely reason, is that the logic of the fighters do not allow the fighters to reach those situations, which means that those transitions are a part of a inaccurate modeling of the system at hand. If the fighters, or the plant model, were modified to better model the actual situation, such anomalies would be reduced. Thus this method establishes $\tilde{\pi}$ values for the available transitions, and it also points out transitions which may not actually have a counterpart in the real scenario, be it from fighters whose logic precludes those situations from happening, or otherwise. The difficulty in creating a plant model for a given system makes such capability in this method algorithm a bonus. After refining the fighter model, plant model, and scaling, we have a usable $\tilde{\pi}$ table, on which we can use the language measure.

	a	b	c	d	e	f	g	h	i	j	k	l
1			.09						.91			
2											1	
3	.44	.01	.02	.1	.006			.11			.01	.3
4	0		0	0				0			0	0
5	.41		.05	.11				.19			0	.24
6			0	0			.64				.36	
7	0		0	.69							.31	
8		.003	.003		.1		.83	.02			.04	
9			0	.32		.67	.01				0	
10	.58		0	.16		.18					.08	
11			.27							.73		
12	.99		.006					.003			0	0
13			.001	.99								
14												
15			0								1	

Table 3: $\tilde{\pi}$ Matrix

Experiment 2: Robustness of measure and optimality algorithm

The scenario developed for the simulation experiments in this setting deployed a fighter airplane against a target that attempts to defend itself by shooting down the fighter. The experiment was conducted using four different but similar plant models and four increasingly aggressive controllers [RW87]. A nominal plant model was developed representing a single fighter aircraft on a mission to destroy a single defensive target (e.g., an anti-aircraft artillery site), as seen previously in prior presentations. From this plant, 3 other models were created through small variations in the $\tilde{\pi}$ values of transitions and X values of the marked states. These variations allow the experiment to examine the language measure's validity, and utilize the X vector feature of assessing the relative worth of marked states. The four models of the plant aircraft are described below:

Plant model 1: The nominal plant model, where the values of the $\tilde{\pi}$ matrix were assigned by their likelihood of occurrence based on limited available data. The X values were set to equal magnitude weights for the two main outcomes of fighter destruction and target destruction. Specifically, X values assigned to these states: aborting the mission (-.05), target destruction (1), and fighter destruction (-1.0). All other states, being unmarked, have a X value of zero.

Plant model 2: Same X values as in Plant model 1, but the $\tilde{\pi}$ values corresponding to the controllable events at each state are equal and their sum unchanged. The remaining $\tilde{\pi}$ values are not altered.

Plant model 3: Same $\tilde{\pi}$ values as those in Plant model 1, but different X values to represent the preference of preserving the fighter over target destruction. Specifically, target destruction has a X value of .95, while fighter destruction has a X value of -1. The remaining X values are the same as those in Plant model 1.

Plant model 4: Same $\tilde{\pi}$ values as in Plant model 2, and X values as in Plant model 3.

The differences between the values of the various plant models are small, between 0.01 to 0.2. The states and events used to represent the fighter-target scenario are similar to the ones from the $\tilde{\pi}$ value generation experiment.

Four supervisors were designed from four simple specification sets, which are described below in order of increasing aggressiveness:

Specification set 1: Only attack if there are no problems with the fighter. That is, no damage (major or minor), and ample fuel. Fighter shall not start attacking if there are any problems, and if a problem (including running low on fuel) develops during attack, the mission will be aborted. Mid-air refueling is not allowed during attack.

Specification set 2: Attack at least once, but then abort if damaged at all. Continue to attack otherwise. If the fighter already has damage, it will attack once and then abort. If the fighter is undamaged when attack begins, it shall abort once damage has taken place. No specifications with respect to fuel levels.

Specification set 3: Abort the attack if the fighter has major damage, and attack at most twice with minor damage,. Otherwise attack until the target is destroyed or weapons have run out. (A new state is added to handle counting in the case of minor damage incurred.) If the fighter already has major damage before attacking, it will abort and not attack. If the fighter incurs major damage while attacking, the mission shall be aborted. If the fighter has minor damage prior to attacking, it will attack at most twice and then abort. If the fighter incurs minor damage during attack, it will attack once more, then abort. No restrictions based on fuel level.

Specification set 4: Attack regardless of problems until target is destroyed or all appropriate weapons have been used. Abort if all available weapons have been used.

Supervisors were designed for the open-loop plant models based on the above four specifications and were labeled according to their specification set number.

For controllers 1, 2, and 4, supervisor construction and application reduces to simple removal of states and transitions from the open-loop plant model. For controller 3, an additional state was added to handle the counting required when considering what the fighter should do when it has incurred minor damage. If it is not obvious how to add/delete states and events to/from the open plant model when applying a controller, product construction is used and the language measure is then applied to that construction.

Results

Language measurements were carried out using each of the four controllers on each of the four plants, as well as the unsupervised version of the plants, for a total of 20 measurements. Table 4 lists the results of these measurements. The No Controller row shows the language measure of the open plant itself, without a controller being applied to it. Those figures are the baseline for comparison of controllers that are applied to those models. Controllers that measure less than the open plant model's measure are considered to be bad or useless controllers (and thus their specifications detrimental): it would be better to attempt the scenario without those controls being applied, for there is a better chance of success according to the measure.

Table 4: Results of the 20 measurements

	Open Plant Models			
No Control	.09898	.1328	.07544	.106
Controller 1	.0797	.104	.0667	.08972
Controller 2	.07698	.09547	.06425	.08054
Controller 3	.10311	.13435	.08422	.1131
Controller 4	.09424	.1175	.08167	.1034

According to the table, controller 3 was the only controller that managed to improve upon all four of the open-loop plant models. As such it consistently was the best control specifications to use for the scenario of the four that were proposed. The consistency is important, because the measure's validity depends on being consistent even over small perturbations of plant models like the changes that were in this experiment. Furthermore, the order of goodness according to the measure was also consistent across the four varieties of plant models: controller 3 was best,

followed by 2, 1, and lastly 4. This observation further establishes the consistency of the measure. If the ranking of controllers could be permuted when they differed by non-trivial amounts as the result of small perturbations in the plant model, the measure would lose validity.

Controllers 1 and 4 were both bad strategies, they consistently yielded measures worse than that of the unrestricted plant, while the performance of controller 2 was mixed; its measures were so close to those of the original plant that its measure compared to the original plant was very sensitive to the changes of the X vector and Π-matrix. The all-or-nothing strategies failed to rank as highly as the strategies that were middling in their aggressiveness, controller 1 being too risk averse and controller 4 allowing for too much risk. The results of the experiment suggest that over the course of repeated runs of this scenario, using the range of parameter perturbation in this experiment), controller 3 would yield the highest percentage of successful runs, followed by 2, 1 and 4.

Now we compare these results to those of applying optimal control onto all of the previous measurements in Table 5.

Table 5: Results from optimal algorithm

w/ Optimal	Open Plant Models			
Open Plant	.12298	.16143	.10286	.13618
Controller 1	.0797	.104	.0667	.08972
Controller 2	.09289	.11545	.08132	.10219
Controller 3	.10564	.13573	.08844	.11365
Controller 4	.09514	.11847	.08257	.10438

As expected, unconstrained optimal control yields the best results. Controller 1's performance is unchanged after optimization, which means that it was already optimal for the set of specifications it was covering. Controller 2's performance has a significant increase, instead of being the worst, it is now better than controller 1.

4.0 Conclusions

With the process of determining $\tilde{\pi}$ values from the first experiment, we have enabled the language measure to be applicable on any system that can be represented as a discrete event plant. The Boeing Simulator was used as a model of the real world, and events can be extracted from it, or other simulations, or even the real world, through the process detailed above. The first experiment helps to shape the plant model and also provides accurate $\tilde{\pi}$ values to use with that plant model. Given those results, one can apply the language measure on control specifications to determine which specifications are best. Alternatively one can use the optimality algorithm to find the best set of enabled and disabled controllable events under the given set of specifications for the controller.

5.0 References

[KG95(1)] R. Kumar and V.K. Garg, *Modeling and Control of Logical Discrete Event Systems,* Kluwer Academic, 1995.

[NS82] A. W. Naylor and G.R. Sell, *Linear Operator Theory in Science and Engineering*, Springer-Verlag, New York, 1982.

[RP02] A. Ray and S. Phoha, "A language measure for discrete-event automata," *International Federation of Automatic Control (IFAC) World Congress b'02*, Barcelona, Spain, 2002.

[RW87] P.J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control and Optimization* 25 (1987), no. 1, 206-230.

[SL98] R. Sengupta and S. Lafortune, "An optimal control theory for discrete event systems," *SIAM J. Control and Optimization* 36 (1998), no. 2, 488-541.

[WR02] X. Wang and A. Ray, "Signed real measure of regular languages," *American Control Conference*, Anchorage, Alaska, 2002.